Understanding Community Effects on Information Diffusion

Shuyang Lin[1(✉)], Qingbo Hu[1], Guan Wang[3], and Philip S. Yu[1,2]

[1] Department of Computer Science,
University of Illinois at Chicago, Chicago, IL, USA
slin38@uic.edu
[2] Institute for Data Science, Tsinghua University, Beijing, China
[3] LinkedIn, Mountain View, CA, USA

Abstract. In social network research, community study is one flourishing aspect which leads to insightful solutions to many practical challenges. Despite the ubiquitous existence of communities in social networks and their properties of depicting users and links, they have not been explicitly considered in information diffusion models. Previous studies on social networks discovered that links between communities function differently from those within communities. However, no information diffusion model has yet considered how the community structure affects the diffusion process.

Motivated by this important absence, we conduct exploratory studies on the effects of communities in information diffusion processes. Our observations on community effects can help to solve many tasks in the studies of information diffusion. As an example, we show its application in solving one of the most important problems about information diffusion: the influence maximization problem. We propose a community-based fast influence (CFI) model which leverages the community effects on the diffusion of information and provides an effective approximate algorithm for the influence maximization problem.

1 Introduction

For many years, community study is one of the hot topics in social network research. Studies in this area offer insightful solutions to many classic problems of social network research, such as network evolution [14], recommendation system [19], and expert finding [2]. Communities can be potentially helpful for studies on diffusion of information in social networks. Previous studies found that links between communities function differently from those within communities: friends in the same communities have stronger links, but weaker links between friends in different communities are crucial in the diffusion of novel information, because these links provide more useful information to people [1,7,12].

Some key problems in the studies of information diffusion have been found difficult to solve by traditional information diffusion methods. Studies on the community structure of social networks may bring new ideas for solving these

© Springer International Publishing Switzerland 2015
T. Cao et al. (Eds.): PAKDD 2015, Part I, LNAI 9077, pp. 82–95, 2015.
DOI: 10.1007/978-3-319-18038-0_7

problems. For example, one of the key problems, the influence maximization problem for the independent cascade model has been proved to be NP-hard [15]. By considering the effects of community studies on the diffusion of information, we can easily come up with some intuitive heuristics to solve that problem more efficiently. For example, we may utilize the community homophily to quickly estimate the influence of users. We may also select seed nodes from different communities to minimize the overlap and maximize the influence.

However, few existing work explicitly studied the effects of communities on the diffusion of information, or use these effects to solve diffusion-related problems. Motivated by this important absence, we introduce the first exploratory study on the effects of communities on information diffusion processes. By analyzing real-world datasets, we study the diffusion of information with communities. We first observe the action homophily of communities, and then introduce the concept of role-based homophily of communities, which consists of influencee role homophily and influencer role homophily. We discover that the role-based homophily is significantly stronger than the action homophily.

Our findings on community effects can lead to insightful solutions to many problems in information diffusion studies. As an example application of these findings, we propose an approximate solution for the influence maximization problem. We design a community-based fast influence (CFI) model based on the influencee role homophily of communities. The CFI model applies a community clustering method to social networks, and makes aggregations on users' roles as influencees. Influence maximization algorithm based on the CFI model can efficiently select seed nodes to maximize the influence. The main contributions are summarized in the following:

1. We conduct quantitative analyses on real-world datasets to explore the effects of community on the diffusion of information.
2. We get valuable findings about the community effects from quantitative analyses. We introduce the concept of role-based homophily of communities. These understandings can bring new insights to the studies of information diffusion.
3. We show an example application of our findings on the influence maximization problem. We propose a community-based fast influence (CFI) model, and an efficient approximate influence maximization algorithm based on that model.

2 Related work

Information Diffusion Problem. Several models have been proposed for the information diffusion processes. The independent cascade (IC) model and its variants are most widely used information diffusion models [10,15,16,21]. The basic idea of the IC model is: if a node in a social network becomes active, it can make its neighbors active with a probability, and for each node the attempts of its neighbors to activate it are independent. The influence maximization problem has been defined for the IC model and a few other information diffusion models [15]. Given an IC model, the problem is to select a seed set with k nodes so that the expected number of active nodes are maximized. This problem has

been proved to be NP-hard. The first solution to it is a greedy algorithm that repeatedly invokes a computational expensive sampling method [15]. Heuristic algorithms and optimized versions of the greedy algorithm have been proposed in previous works [3,4,17]. Work in [22] proposed an heuristic algorithm which finds influencers from communities. Different from that work, our proposed model is built on observations on real data and baed on a substantially different idea. A recent work in [8] defined a group-based version of the influence maximization problem. The predefined groups studied in that work were not conceptually equivalent to the communities studied in this paper.

Community Detection. Community detection in social networks has been studied for years. Varieties of algorithms have been proposed. A good survey is available in [18]. We are not going to discuss the varieties of existing community detection methods, except for those that are related to our work in this paper. Modularity-based methods are a major class of community detection methods. Among these methods, the fast greedy method [5] is frequently used for community detection on large-scale networks. In [20], Rosvall et al. proposed the infomap method. Substantially different from modularity-based methods, the infomap method is based on flows carried by networks [20]. The SHRINK algorithm in [13] is another algorithm that is related to our work. It is a parameter-free hierarchical network clustering algorithm that combines the advantages of density-based clustering and modularity optimization methods. Work in [23] utilized social influence modeling methods in the detection of communities.

3 Preliminary

3.1 Notations

A **social network** $G = \{V, E\}$ is a directed graph with a node set V and an edge set E. A node $v_i \in V$ represents a user in the social network, while a directed edge $e_{ij} \in E$ represents a link from v_i to v_j.

A **community** C in the social network G is a subset of the node set V. We consider non-overlapping communities in this paper. In other words, we consider the partition of V into a set of communities $\mathcal{C} = \{C_i\}_{i=1}^{m}$. Each user in the network should belong to exactly one of the communities in $\mathcal{C}$. Given a graph G, a **community detection** algorithm divides the graph G into a set of communities $\mathcal{C}$. There are a lot of different community detection algorithms. Generally, a good community detection algorithm finds a partition, so that (1) each community is a relatively independent compartment of the graph, and (2) nodes in the same community tend to have dense links between each other.

We follow the definition of **information diffusion process** in the IC model [15]. An information diffusion process starts with a set of seed nodes that are active at the first place. Active nodes can activate their out-neighbors in the social network. Once a node is activated, it becomes active and can never become inactive again. It is quite often for real applications that the information diffusion processes cannot be directly observed. For example, we may observe that a

person got infected by influenza, but we do not know from whom he got infected. We define a **cascade** $O = \{(v_1, t_1), \ldots, (v_m, t_m)\}$ as the set of user actions during an information diffusion process. An action (v_i, t_i) in O represents that the user v_i becomes active at time t_i. In this paper, we focus on the scenario that the information diffusion processes are not directly observed, but a set of cascades is observed.

3.2 Datasets

Foursquare[9]. In this dataset, nodes represent users of the Foursquare website, while edges represent friendship relations. Actions are defined by check-ins of users. Each cascade corresponds to a location. When a user checks in at a location for the first time, she becomes active for the corresponding cascade. This dataset contains 18,107 users, 245,034 friendship relations, and 476,482 actions of 43,063 cascades.

DBLP. In this dataset, nodes represent authors, while edges represent co-author relations. We extract a subgraph of the DBLP network with authors and papers in the areas of data mining and machine learning. We define cascades by terms (defined by bi-grams) in the titles of papers. When an author has a paper with a certain term in the title for the first time, he becomes active for the corresponding cascade. This dataset contains 6,896 users, 111,044 friendship relations, and 1,655,778 actions of 162,904 cascades.

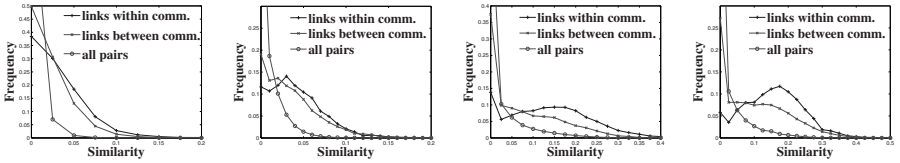
4 Observations

In this section, we explore the community effects on information diffusion processes via analyses on real-world datasets. We first identify communities in social networks, and then study cascades with respect to these communities.

4.1 Identifying Communities for Information Diffusion

Communities in social networks can be defined in many different ways. To understand the effects of communities on the information diffusion in general, we apply two different community detection algorithms to the two networks, and conduct community effect analyses for both algorithms.

The two community detection methods that we use to identify communities are the fast greedy (FG) method [5], and the infomap (IM) method [20]. The FG method is based on the well-adopted idea of modularity maximization, while the IM method is a flow-based method, which is essentially different from the modularity maximization methods. We choose these two methods because (1) they are all widely-used community detection methods that prove to be efficient and accurate, and (2) they are based on substantially different ideas. Both methods are implemented in the igraph network analysis package [6].



(a) DBLP, IM (b) DBLP, FG (c) Foursquare, IM (d) Foursquare, FG

Fig. 1. Distribution of similarity between actions of user pairs

4.2 Action Homophily of Communities

We first look into the effects that communities have on the actions of users. We construct a vector for each user to keep the action information of that user, and then compare the vectors between pairs of users. We check whether the users who belong to the same community are more likely to have similar actions.

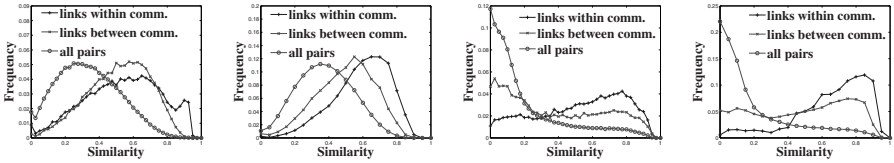
Given a set of cascades $\mathcal{O} = \{O_1, \ldots, O_m\}$, we define an action vector $\mathbf{a_i}$ for each user v_i, where $\mathbf{a_i} = (a_{i0}, \ldots, a_{im})$. If the user v_i has an action in the cascade O_j, we set a_{ij} to 1. Otherwise, we set a_{ij} to 0. For each pair of users v_i and v_j, we calculate the cosine similarity between the action vectors $\mathbf{a_i}$ and $\mathbf{a_j}$, and then study the distribution of similarity. We consider three different cases here: (1) There is an edge e_{ij} between v_i and v_j, and v_i and v_j belong to the same community; (2) There is an edge e_{ij} between v_i and v_j, but v_i and v_j belong to different communities; (3) v_i and v_j is an arbitrary pair of nodes, may or may not having an edge between them. For each case, we plot the distributions of similarity, and check whether there is any difference between the distributions.

Figure 1 shows the distributions of similarity in two datasets, with two sets of communities in each dataset. In each setting, we observe a similar discrepancy among the three distributions: comparing with linked pairs in different communities, linked pairs in the same communities have larger similarity; comparing with arbitrary pairs, linked pairs have much larger similarity. Intuitively, friends in the same communities tend to have stronger link between each other, and they have more chances to influence each other indirectly via common friends. The results are quite consistent for different community detection methods.

4.3 Role-Based Homophily of Communities

We have observed the action homophily of communities. However, although the similarity between linked pairs in the same communities is relative larger than the similarity in the other two cases, it is still quite small (typically, less than 0.3). In this section, we introduce the role-based homophily of communities, and show that the role-based homophily is more significant than the action homophily.

With a set of cascades $\mathcal{O}$, we build a support matrix S for the influence between users in the social networks. The element at the i-th row and the j-th column of the matrix S is the number of potential influences from the user v_i to the user v_j. We say there is a potential influence from v_i to v_j, if both of them have actions in the same cascade, and the time of v_i's action is earlier



(a) DBLP, IM (b) DBLP, FG (c) Foursquare, IM (d) Foursquare, FG

Fig. 2. Distribution of similarity between influencer feature vectors of user pairs

than the time of v_j's action. Formally, it is defined as: $s_{ij} = \mid \{O_k \in \mathcal{O} \mid v_i, v_j \in V(O_k) \wedge t_i^{O_k} < t_j^{O_k}\} \mid$, where $V(O_k)$ is the set of users that has an action in the cascade O_k, and $t_i^{O_k}$ is the time of v_i's action in the cascade O_k.

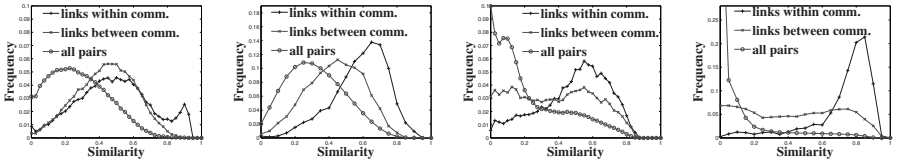
We define $\mathbf{s}_{i*}$, the i-th row of S, as the **influencer feature vector** of v_i, and $\mathbf{s}_{*i}$, the i-th column of S, as the **influencee feature vector** of v_i. The influencer feature vector $\mathbf{s}_{i*}$ captures the influence that v_i has on other users in the social networks, while the influencee feature vector $\mathbf{s}_{*i}$ captures the influence from other users to the user v_i.

Similar to what we did in Section 4.2, we calculate the cosine similarity between the influencer/influencee feature vectors, and compare the distributions. Figures 2 and 3 show the comparison of distributions of influencer feature vector and influencee feature vector, respectively. Similar to Figure 1, comparing with the other two cases, the similarity is larger for the case that users are linked and are in the same communities. There are a few new observations that are interesting:

First, distributions of similarity between influencer/influencee feature vector (Figures 2 and 3) show significantly larger discrepancy than the distributions of similarity of action vector (Figure 1). This observation suggests that for users in the same communities the role-based homophily is much stronger than the action homophily. The effect of community in the information diffusion process is better reflected by the roles that users play in the information diffusion process, rather than the results of information diffusion process (whether being active or inactive for a cascade).

Second, for friends in the same community, the similarity value of influencer and influencee feature vectors (typically larger than 0.5) is larger than the similarity of action vectors (typically less than 0.3). It suggests that aggregation on the influencer/influencee feature vectors of users without significant loss of accuracy is more feasible.

Third, the influencee-based homophily is more significant than the influencer-based homophily, especially for the FG algorithm. This is easy to understand by the following example: professors and students in the same research lab have similar behaviors as influencees, because when a cascade reaches anyone in the lab, it is very likely that cascade will reach everyone in the lab quickly, but professors are probably much stronger influencers than students.



(a) DBLP, IM (b) DBLP, FG (c) Foursquare, IM (d) Foursquare, FG

Fig. 3. Distribution of similarity between influencee feature vectors of user pairs

5 Community-Based Fast Influence Model

Based on the observations in Section 4, we are able to design an efficient influence model which makes use of the community effects. The community-based fast influence (CFI) model we propose in this section is an approximate model for the IC model. The whole framework has three components, namely influence decoupling, community detection, and influence maximization.

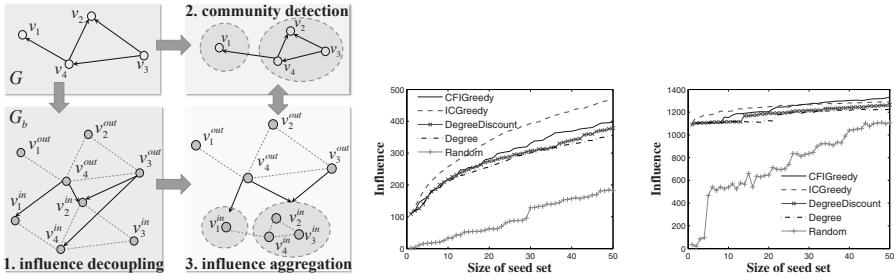
5.1 Influence Decoupling

An intuitive way to construct an approximate information diffusion model based on community effects is to consider each community as a "super-node" and make information propagates through "super-edges" between "super-nodes". The coarse-grained information diffusion model in [8] is based on a similar idea.

Although this intuitive model is simple and seems reasonable, it may not work for our task here. When we consider a community as a "super-node", we have to aggregate users' roles as influencers as well as users' roles as influencees. This may cause a problem: the influence maximization problem requires us to determine how influential each user is and find the set of seed nodes that maximizes the influence. When we aggregate the roles of users as influencers, we lose the necessary information for solving the influence maximization problem.

To avoid this problem, the CFI model considers the roles of users as influencers and influencees separately. To be specific, we split each node v_i in the network G into an influencer node v_i^{out} and influencee node v_i^{in}, and transform the network into a bipartite graph G_b. In the graph G_b, there is an edge from v_i^{out} to v_j^{in} if and only if the edge e_{ij} exists in the original graph G. We call this transformation from the original network G to the bipartite graph G_b **influence decoupling**. The left part of Figure 4(a) shows an example of influence decoupling. In the original graph G, there is an edge from v_4 to v_1. Correspondingly, there is an edge from v_4^{out} to v_1^{in} in G_b. The result of influence decoupling is that we can apply the community-based aggregation to the influencee nodes only.

If we apply the original IC model directly to the decoupled graph G_b, we will end up with cascades with only two levels, i.e. only the nodes that are direct out-neighbors of the seed nodes can become active. This problem can be approximately solved by the community detection and the aggregation of influencee nodes. Instead of limiting influence to direct out-neighbors, the CFI



(a) (b) DBLP (c) Foursquare

Fig. 4. (a) Inference of the CFI model. (b)-(c) Influence of different sizes of seed set

model allows users to have influence on the communities that their direct out-neighbors belong to. Notice that we do omit the indirect influence from a user to the nodes that are neither his out-neighbor nor in the same communities as his out-neighbor. This is indeed a trade-off between the accuracy and efficiency, but the loss of accuracy is actually negligible. This is because the influence between nodes in different communities are smaller than the influence between nodes in the same communities, and indirect influence are generally very small. We will also show by experiment that the CFI model is a good enough approximation to the original IC model.

5.2 Identifying Communities

We now discuss the community detection algorithm. As we have discussed in the last section, users in the same community should be similar influencees. To identify communities so that users in the same communities are similar as influencees, we design an agglomerative clustering algorithm. It starts with clusters with single users, and iteratively merges clusters together based on similarity between clusters. As shown in Figure 4(a), the clustering procedure is conducted on the original graph G, but the similarity is defined by users' roles as influencees, and the communities detected by the algorithm will finally be applied to the influencee nodes in the decoupled graph G_b.

Similarity. The similarity between two clusters is defined as the cosine similarity between their incident influence probability vectors. Let $p_{i\to j}$ be the probability that v_i influences v_j directly or indirectly (i.e. the probability that v_j becomes active if v_i is the single seed node). For a cluster $C = \{v_{i_1}, \dots v_{i_{n_C}}\}$, we define the influence that user v_j on C as:

$$q_{j\to C} = \begin{cases} \frac{1}{n_c}\sum_{k=1}^{n_C} p_{j\to i_k} & \text{if } e_{j,i_k} \in E \text{ for some } i_k \in C \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

where n_C is the number of nodes in the cluster C.

We define incident influence probability vectors of community C as $\mathbf{q}_C = (q_{1\to C}, \dots q_{n\to C})$, and the similarity between two clusters C_1 and C_2 as $sim(C_1, C_2) = \mathbf{q}_{C_1} \cdot \mathbf{q}_{C_2}/(\|\mathbf{q}_{C_1}\|\|\mathbf{q}_{C_2}\|)$.

Influence Probability Estimation. Similar to the learning algorithm for the IC model in [10], given a set of cascades $\mathcal{O}$, we estimate the influence probability $p_{i\to j}$ from cascades by the equation as follows:

$$\hat{p}_{i\to j} = \frac{s_{ij}}{s_i} = \frac{\mid \{O_k \in \mathcal{O} \mid v_i, v_j \in V(O_k) \wedge t_i^{O_k} < t_j^{O_k}\} \mid}{\mid \{O_k \in \mathcal{O} \mid v_i \in V(O_k)\} \mid} \tag{2}$$

Since that v_i becomes active earlier than v_j does not necessarily imply that v_i directly or indirectly influences v_j, $\hat{p}_{i\to j}$ is not an unbiased estimator of $p_{i\to j}$. However, it is still a good enough estimator for the CFI model.

Community Detection and Influence Aggregation. The purpose of community detection in the CFI model learning is to aggregate users who play similar roles as influencees, while keep the accuracy of the original IC model. To serve this purpose, we adopt a community detection strategy that is similar to the algorithm in [13]. By iteratively merging clusters into larger one, we get a sequence of super-graph $G_0, G_1, G_2, \ldots$. Each node in these super-graphs corresponds to a cluster. The algorithm starts with graph G_0, in which each cluster contains a single user. In each step t, we find from G_t connected subgraphs that contain similar nodes, and merge these subgraphs to generate a new super-graph G_{t+1}. We repeat these steps, until the similarity between any two neighbors in G_t are below a threshold θ.

Let $\mathcal{C}^{(t)} = \{C_1, \ldots C_{m^{(t)}}\}$ be the set of clusters at the t-th iteration. We say two clusters C_1 and C_2 are neighbors, if there exist $v_i \in C_1$ and $v_j \in C_2$, s.t. edge e_{ij} or e_{ji} exists. For a pair of connected clusters, we say they are a **mutually most similar pair (ms-pair)** with similarity ϵ (denoted by $C_1 \leftrightarrow^{\epsilon} C_2$), if $\epsilon = sim(C_1, C_2) = \max_{C_i \in N(C_1)} sim(C_1, C_i) = \max_{C_i \in N(C_2)} sim(C_2, C_i)$, where $N(C_i)$ is the set of neighbors of C_i.

We define a **ms-subgraph** as a maximal connected subgraph of G_t that are connected by ms-pairs. Formally, a graph D is a ms-subgraph of G_t with similarity ϵ if and only if (1) for any two nodes $C_i, C_j \in D$, there exist a path $< C_i, C_1 \ldots C_k, C_j >$ in D, s.t. $C_i \leftrightarrow^{\epsilon} C_1$, $C_1 \leftrightarrow^{\epsilon} C_2$,$\ldots$,$C_{k-1} \leftrightarrow^{\epsilon} C_k$, $C_k \leftrightarrow^{\epsilon} C_j$; (2) for any nodes $C_i \notin D$ and $C_j \in D$, C_i and C_j are not a ms-pair. By this definition, the graph can be partitioned into ms-subgraphs (some ms-subgraphs may contain only one single node). By merging ms-subgraphs into new nodes, the original super-graph can be reduced into a smaller super-graph.

At the iteration t, we first find out all the ms-subgraphs of G_t, and then merge each ms-subgraph D that contains more than one nodes and has similarity $\epsilon \geq \theta$ into a new node. The new node is a neighbor to any node that was a neighbor of any node in D, and the similarity between the new nodes and its neighbors need to be recalculated. The algorithm stops when the similarity between each linked nodes are less than the threshold θ, and the clusters at that point of time are taken as communities.

5.3 CFI-Based Influence Maximization Algorithm

In this subsection, we show how we can design a CFI-based algorithm for the influence maximization of the IC model. The influence maximization problem

is defined as follows: Given an IC model and an integer $k > 0$, find a set of k seed nodes, so that the influence of the seed nodes is maximized. The standard method to solve this problem is a greedy algorithm [15]. It starts with finding one seed node that maximizes the influence, and then adds a second node to the seed node set so that the increase of influence is maximized. In this way, it repeatedly adds nodes to the seed node set, until it gets k seed nodes. This greedy algorithm is very time-consuming, because in each step it uses the Monte Carlo method to evaluate all the remaining nodes. Optimized versions of the greedy algorithm have been proposed in [17] and [11], and heuristic algorithms have been proposed in [3]. These algorithms also use sampling for the evaluation of nodes. We can get a new heuristic algorithm based on the CFI model. This new heuristic algorithm does not involve random sampling, so it is faster than the existing algorithms, especially when the number of seed nodes k is large.

The CFI-based influence maximization algorithm also adopts a greedy framework. In each step t, the node that can maximize the influence increase is selected. The problem is how we can estimate the influence increase using the CFI model. When $t = 1$, the problem is reduced to estimating the influence of each single node. Let $\mathcal{C} = \{C_1, \ldots C_m\}$ be the set of communities in the CFI model. We estimate the influence of a user v_i as $Inf(\{v_i\}) = \sum_{C \in \mathcal{C}} n_c q_{i \to C}$, where n_c is the number of users in the community C.

Once we select the node with the greatest influence to be the first seed node, we cannot simply select the second most influential node to be the second seed node, because the nodes activated by the first node and the second node may overlap. We need to deduct the number of nodes that has already been activated by the first node. To do that, we decrease the number of nodes from each community by the estimated influence of the first node v_{i_1}. Formally, we let $n_c^1 = n_c - n_c q_{i_1 \to C}$, which is the expected number of nodes in the community C that are not activated by the influence of v_{i_1}, and then we select the second node v_{i_2} by maximizing the increase of influence: $\Delta Inf(v) = Inf(\{v_{i_1}, v\}) - Inf(\{v_{i_1}\}) = \sum_{C \in \mathcal{C}} n_c^1 q_{i_2 \to C}$. For $t = 3, \ldots, k$, we can repeat the above step to select $v_{i_3}, \ldots, v_{i_k}$. Generally, we select v_{i_t} by maximizing $\sum_{C \in \mathcal{C}} n_c^{t-1} q_{i_t \to C}$, where $n_c^{t-1} = n_c^{t-2} - n_c^{t-2} q_{i_{t-1} \to C}$.

6 Experiment

6.1 Experiment Setup

We use the DBLP and Foursquare networks described in Section 3 for the experiment. For each network, we construct an IC model by assigning diffusion probability $1 - e^{-0.01c}$ to each edge. A similar method for model construction has been used in [4]. For the DBLP network, c is the number of papers coauthored by the two authors. For the Foursquare network, c is the number of locations that both users visited. We do not construct the ground-truth models by learning the diffusion probabilities directly from the actions in the datasets because we want to avoid the inaccuracy caused by model learning algorithm.

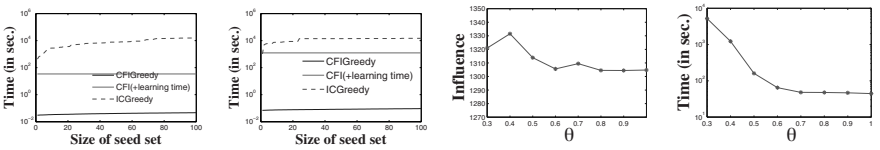
We then sample each ground truth IC model to get 5,000 cascades, each with 10 seed nodes and use the sampled cascades to learn the CFI model. For the baselines, since they are all based on the IC model, we directly apply the influence maximization algorithms on the ground truth model. We evaluate the influence of seed nodes by sampling the ground truth model 10,000 times to get the average number of active nodes. We compare the following algorithms:

– **CFIGreedy** The CFI-based influence maximization algorithm with $\theta = 0.4$.
– **ICGreedy** The greedy influence maximization of IC model with the CELF++ optimization [11]. We take a sample size of $10,000$ to estimate the influence.
– **Degree** The heuristic algorithm that selects the nodes with the largest weighted degree. The weighted degree of a node is the sum of the diffusion probabilities over the out-going edges.
– **DegreeDiscount** The degree discount heuristics based on the degree heuristics [4]. The basic idea is to discount the degree for users whose friends have been selected as seed nodes.
– **Random** Randomly selecting seed nodes.

6.2 Results

Effectiveness Results for Influence Maximization. First, we present the effectiveness results of the influence maximization algorithms in terms of the number of seed nodes. We test the effectiveness of each algorithm with increasing number of seed nodes. The results of the DBLP and Foursquare datasets are illustrated in Figures 4(b) and 4(c), respectively. In each case, we illustrate the number of seed nodes on the X-axis, and the influence of seed nodes on the Y-axis. For the Foursquare data, $CFIGreedy$ performs worse than $ICGreedy$ when the size of the seed set is small, but does better than $ICGreedy$ when the size is greater than 25. This is a very interesting observation. Although the CFI model is designed to be an approximate model for the IC model, the greedy algorithm of the CFI model does not necessarily performs worse than the greedy algorithm of the IC model. This is because the CFI model considers the community structure of social networks, and the consideration of community structure may favor combinations of seed nodes that cover more communities. For the DBLP dataset, $CFIGreedy$ is less effective than $ICGreedy$. However, the difference is not very significant, especially when we consider the fact that $CFIGreedy$ is significantly faster. Besides, $CFIGreedy$ consistently outperforms $DegreeDiscount$, $Degree$ and $Random$. Notice that although $DegreeDiscount$ is a simple heuristic method, previous work showed that it is a very effective method that nearly matches the performance of $ICGreedy$ [4].

Efficiency Results for Influence Maximization. We also tested the efficiency of influence maximization methods with varying number of seed nodes. The efficiency results for the DBLP and Foursquare datasets are illustrated in Figures 5(a) and 5(b), respectively. The X-axis denotes the number of seed nodes, whereas the Y-axis denotes the running time. Since heuristics as $Random$,



(a) DBLP (b) Foursquare (c) Influence (d) Runing time

Fig. 5. (a)-(b) Running time with different sizes of seed set; (c)-(d) Effects of θ

Degree, and *DegreeDiscount* are obviously very fast, we only show the running time of $CFIGreedy$ and $ICGreedy$. As illustrated in the figures, influence maximization based on $CFIGreedy$ is several orders of magnitudes faster than $ICGreedy$ with CELF++ optimization. We also add together the time spent on the learning of the CFI model and the running time of $CFIGreedy$ to get a total time for the influence maximization on the CFI model, and illustrate the total time as "CFI(+learning time)" in the figures. Even when the learning time is added, the total running time for the CFI model is still significantly smaller the running time of $ICGreedy$. For example, for the DBLP dataset, it takes $ICGreedy$ 9,079 seconds to find 60 seed nodes, while the total running time of the CFI model is 34 seconds. Notice that, in real applications, the IC models also need to be learned from user actions, and the running time of $ICGreedy$ should also be added with the learning time of the IC model.

Parameter Sensitivity. Finally, we tested the sensitivity of the CFI-based influence maximization with the clustering threshold θ. Figure 5(c) shows the influence of seed nodes selected by the CFI model with varying θ. We illustrate the value of θ on the X-axis, and the influence of seed nodes on the Y-axis. Figure 5(d) shows the total running time of influence maximization with varying θ. We illustrate the value of θ on the X-axis, and the total running time of the influence maximization (the running time of model learning plus the running time of $CFIGreedy$) on the Y-axis. In each case, the number of seed nodes is set to 50. We show the results on the Foursquare dataset, while similar trends are observed on the DBLP dataset. When the threshold θ decreases, the running time increases, because the agglomerative clustering takes more steps when θ is smaller. It is an interesting observation that the influence does not monotonically increases when θ decreases. When θ is too large, the size of communities are very small, so the CFI model omits too much indirect influence. When θ is too small, the users in the same community do not have enough similarity between each other. Both cases cause loss of accuracy. Nevertheless, we notice that the variation of influence is not significant. The Y-axis of Figure 5(c) does not start at 0. When θ varies from 0.3 to 1.0, the variation of influence is within $\pm 1.5\%$.

7 Conclusion

In this paper, we explore the effects of communities on the information diffusion processes. We quantitatively analyze the real-world information diffusion

datasets to get insightful findings on the community effects. As an application of these findings, we propose the CFI model, which is substantially different from existing approximate algorithms. Experiment shows that the CFI-based influence maximization algorithm can get comparable effectiveness as influence maximization algorithms based on the IC model, but is significantly faster. Our work sheds light on the effects of communities in the diffusion of information, and brings a new idea to the approximation of information diffusion processes.

Acknowledgments. This work is supported in part by NSF through grants CNS-1115234, and OISE-1129076, Google Research Award, and the Pinnacle Lab at Singapore Management University.

References

1. Bakshy, E., Rosenn, I., Marlow, C., Adamic, L.: The role of social networks in information diffusion. In: WWW (2012)
2. Balog, K., Azzopardi, L., de Rijke, M: Formal models for expert finding in enterprise corpora. In: SIGIR (2006)
3. Chen, W., Wang, C., Wang, Y.: Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: KDD (2010)
4. Chen, W., Wang, Y.: Efficient influence maximization in social networks. In: KDD (2009)
5. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70**(6 Pt 2), 066111 (2004)
6. Csardi, G., Nepusz, T.: The igraph software package for complex network research. Inter. Journal, Complex Systems, 1695 (2006)
7. Eagle, N., Macy, M., Claxton, R.: Network diversity and economic development. Science **328**(5981), 1029–1031 (2010)
8. Eftekhar, M., Ganjali, Y., Koudas, N.: Information cascade at group scale. In: KDD (2013)
9. Gao, H., Tang, J., Liu, H.: Exploring social-historical ties on location-based social networks. In: ICWSM (2012)
10. Goyal, A., Bonchi, F., Lakshmanan, L.V.: Learning influence probabilities in social networks. In: WSDM (2010)
11. Goyal, A., Lu, W., Lakshmanan, L.V.: Celf++: optimizing the greedy algorithm for influence maximization in social networks. In: WWW (2011)
12. Granovetter, M.S.: The Strength of Weak Ties. The American Journal of Sociology **78**(6), 1360–1380 (1973)
13. Huang, J., et al.: Shrink: a structural clustering algorithm for detecting hierarchical communities in networks. In: CIKM (2010)
14. Jin, E.M., Girvan, M., Newman, M.E.J.: Structure of growing social networks. Phys. Rev. E **64**, 046132 (2001)
15. Kempe, D., Kleinberg, J.: Maximizing the spread of influence through a social network. In: KDD (2003)
16. Lappas, T., Terzi, E., Gunopulos, D., Mannila, H.: Finding effectors in social networks. In: KDD (2010)
17. Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., Vanbriesen, J., Glance, N.: Cost-effective outbreak detection in Networks. In: KDD (2007)

18. Newman, M.E.J.: Modularity and community structure in networks. PNAS **103**(23), 8577–8582 (2006)
19. Reddy, P.K., Kitsuregawa, M., Sreekanth, P., Rao, S.S.: A graph based approach to extract a neighborhood customer community for collaborative filtering. In: Bhalla, S. (ed.) DNIS 2002. LNCS, vol. 2544, pp. 188–200. Springer, Heidelberg (2002)
20. Rosvall, M., Axelsson, D., Bergstrom, C.T.: The map equation. The European Physical Journal Special Topics **178**(1), 13–23 (2009)
21. Saito, K., Kimura, M., Ohara, K., Motoda, H.: Learning continuous-time information diffusion model for social behavioral data analysis. In: Zhou, Z.-H., Washio, T. (eds.) ACML 2009. LNCS, vol. 5828, pp. 322–337. Springer, Heidelberg (2009)
22. Wang, Y., Cong, G., Song, G., Xie, K.: Community-based greedy algorithm for mining top-K influential nodes in mobile social networks. In: KDD (2010)
23. Zhou, Y., Liu, L.: Social influence based clustering of heterogeneous information networks. In: KDD (2013)